IN THE SPECIFICATION

Please insert the following paragraph between current paragraphs [0011] on page 3 of the Specification and [0012], spanning pages 3 and 4 of the Specification:

-- In another aspect, a method of protecting a host from unauthorized client access over a network includes the steps of: creating a prover agent application on the client; creating a verifier agent application on the host; and creating a trusted source application to generate and publish encrypted values of a secret and product of first and second large prime numbers. The encrypted values are read for the secret and product, by the provider and verifier from the trusted source. The secret is decrypted, by the prover and verifier, and the product is decrypted, by the prover and verifier. A plurality of verification dialog is performed between the prover and verifier, wherein the prover demonstrates knowledge of the secret and product without exposing the values of the secret and product. The client is denied access when the prover fails to demonstrate knowledge of the secret and product, and granted access when the client succeeds in demonstrating knowledge of the secret and product. --

Please amend paragraph [0024], found on pages 5-6 of the Specification, as follows:

-- FIG. 3 shows a challenge-response-validation iteration dialog between a prover agent (shown as process 48) and a verifier agent (shown as process 50). Process 48 performs processing to establish a need to authenticate and begins zero-knowledge identification protocol 46 in step 52, which may include retrieving and decrypting current values of secret s" and the product n". In step 52, process 48 (prover) sends a signal 54 to process 50 (verifier) to begin zero-knowledge identification protocol 46. In step 56, process 50 (verifier) performs any initial processing, which may include retrieving and decrypting current values of secret s" and product n". In step 56, process 50 sends signal 58 to process 48 (prover) to begin the authorization process. In step 60, process 48 (prover) generates a random number ("r"). Random number r is then used, in step 62, to generate a number x such that x=r{circumflex over ( )}t mod n. In step 62, process 48 sends a signal 64 containing x to process 50 (verifier). In step 66, process 50 (verifier) then calculates a reply value b as a member of set {0 . . . t-1}. In step 66, process 50 sends a signal 68 containing b to process 48 (prover). In step 70, process 48 (prover) uses b to calculate a number y such that y=rs{circumflex over ( )}b. In step 70, process 48 sends a signal

72 containing y to process 50 (verifier). Step 74 in process 50 is a decision. In step 74, process 50 performs a test to determine if process 48 (prover) has passed this iteration of zero-knowledge identification protocol 46. If y{circumflex over ( )}t mod n=(xv{circumflex over ( )}b) mod n and y< >0, then process 50 continues with step 78; otherwise process 50 continues with step 76. Step 78 in process 50 is a decision. In step 78, the number of challenge-response-verification iterations is compared to the number of iterations required to establish a suitable probability of correct authentication. If the number of challenge-response-validation iterations performed is the same as the number of challenge-response-validation iterations required, and process 48 (prover) has not failed any iterations, then process 48 continues with step 82; otherwise process 50 ~~signals~~sends a signal 80 to process 48 to continue with step 60, thus beginning another challenge-response-validation iteration by repeating steps 60 through 74. --

Please amend paragraph [0026], found on pages 6-7 of the Specification, as follows:

-- FIG. 4 shows a system 89 with three clients 90(1-3), each running a prover agent 91(1-3), and a host 92 running an authentication agent 96 (verifier). Prover agents 91(1-3) implement process 48, FIG. 3, for example. Authentication agent 96 implements process 50, FIG. 3, for example. Communication links 100(1-3) establishes connectivity between clients 90(1-3) and a connection module 94 within host 92. In system 89, client 90(1) seeks access to secure area 98 of host 92. Communication link 100(1) establishes connectivity between client 90(1) and connection module 94 within host 92. In one example, communication link 100(1) is a telephone dial-up connection. In another example, communication link 100(2) is an Internet connection. In another example, authentication agent 96 (verifier) protects secure area 98 allowing access only to authenticated clients. Communication link 100(3) is an Ethernet LAN connection. After client 90(1) is authenticated by host (92), a connection 102 is established and client 90(1) is allowed access to secure area 98. Once this connection has been established, authentication agent 96 may distribute a new secret from trusted source 106 to prover agent 90(1) for use in future authentication dialog. When prover agent 90(1) requests authentication at a future time after connection 110 has been broken, authentication agent 96 requests credentials from prover agent 90(1) from trusted source 106 via the hosts internal connection 104. At this point the authentication dialog may take place between client 90(1) and host 92 to reestablish a trusted connection. --

IN THE DRAWINGS

The five attached sheets of drawings includes changes to FIGs. 1-5. These sheets replace the drawings filed 16 October, 2003. No new matter is added.

ATTACHMENTS: FIVE (5) REPLACEMENT SHEETS.